Introduction to security and cryptography
Lecture 1: Old and Modern Encryption

Marius Lombard-Platet

ENS, Almerys

Introduction

Administrative informations

Teacher: Marius Lombard-Platet (me)
marius.lombard-platet@uca.fr

Lessons

- Today
- 23/09
- 25/09
- 01/10
- 03/10

Exam

- 12/11
- 1 hour
- 1 sheet of written notes should be allowed (tbd)

Prerequesites

Not much, basics in math and computer science. More specific techniques will be explained beforehand.

If you do not know something, let me know!

What is this course about?

We will expose the basic notions, techniques, models used in security and cryptography. While this course will help you be familiar with the topic, you will need to take a deep dive into maths, computer science, sociology and so on.

Today

- What is crypto?
- History of crypto
- Modern crypto
- One way functions

Motivation

How do you win a war without fighting it?

How do you win a war without fighting it?

> *O divine art of subtlety and secrecy! Through you we learn to be invisible, through you inaudible; and hence we can hold the enemy's fate in our hands.*

Lao Tseu, *The Art of War*

Cryptography can answer the following requirements:

- **Confidentiality**

Cryptography can answer the following requirements:

- **Confidentiality**: the data is only accessible to authorized people

Cryptography can answer the following requirements:

- **Confidentiality**: the data is only accessible to authorized people
- **Authenticity**

Cryptography can answer the following requirements:

- **Confidentiality**: the data is only accessible to authorized people
- **Authenticity**: the message sender is really who they pretend to be

Cryptography can answer the following requirements:

- **Confidentiality**: the data is only accessible to authorized people
- **Authenticity**: the message sender is really who they pretend to be
- **Non-repudiation**

Cryptography can answer the following requirements:

- **Confidentiality**: the data is only accessible to authorized people
- **Authenticity**: the message sender is really who they pretend to be
- **Non-repudiation**: the sender cannot deny they have sent the message
- **Integrity**

Cryptography can answer the following requirements:

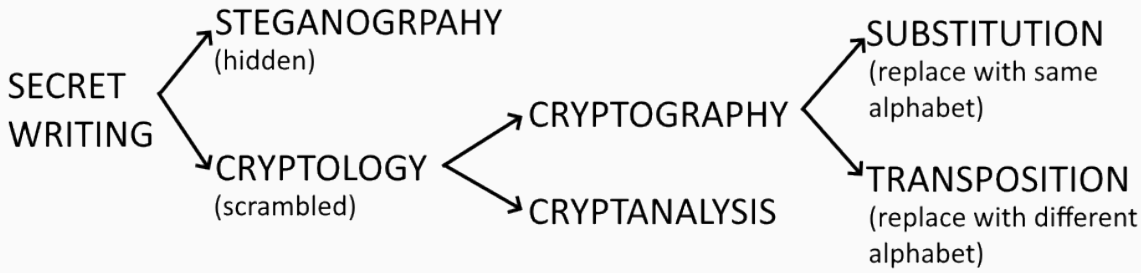
- **Confidentiality**: the data is only accessible to authorized people
- **Authenticity**: the message sender is really who they pretend to be
- **Non-repudiation**: the sender cannot deny they have sent the message
- **Integrity**: The data cannot be altered

Cryptography can answer the following requirements:

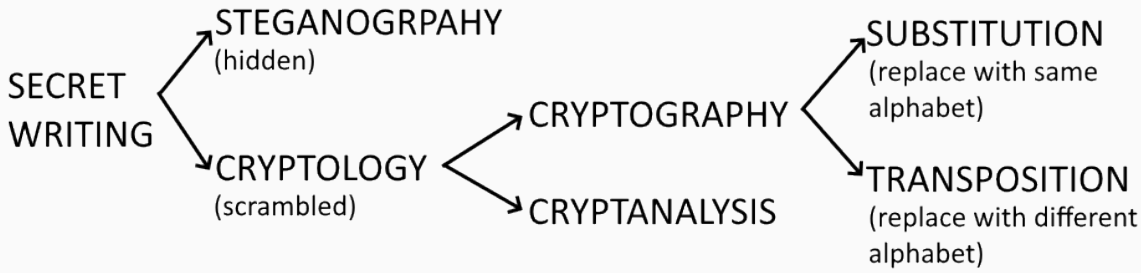
- **Confidentiality**: the data is only accessible to authorized people
- **Authenticity**: the message sender is really who they pretend to be
- **Non-repudiation**: the sender cannot deny they have sent the message
- **Integrity**: The data cannot be altered

And some more.

How to protect information?



How to protect information?



- **Cryptology**: the study of secret writing.
- **Steganography**: the art of hiding messages in other messages.
- **Cryptography**: the science of secret writing.
  Note: terms like **encrypt**, **encode**, and **encipher** are often (loosely and wrongly) used interchangeably

What about steganography?

Steganography is one of the oldest examples of secret writing we know, along with substitution.

First known example: Aristagoras, 500 B.C. Using the shaved skull of a slave, then waiting for the hair to grow back.

What about steganography?

Other examples in History: WWI (telegraphs), WWII (Belgian knitting, British letters)...

Modern solutions are more sophisticated, but still far from perfect (laser printers, covert LANs, LSB...)

What about steganography?

Other examples in History: WWI (telegraphs), WWII (Belgian knitting, British letters)...

Modern solutions are more sophisticated, but still far from perfect (laser printers, covert LANs, LSB...)

Figure 1: LSB steganography (Aaron Miller)

Also see the first step of the ANSSI logo challenge

What about steganography?

However, steganography is mostly craftsmanship, and pretty much by definition cannot be standardised.

What about steganography?

However, steganography is mostly craftsmanship, and pretty much by definition cannot be standardised.

Kerchkhoffs's principle
A good cryptosystem should be secure even though everything, except for the key, is publicly known.

What about steganography?

However, steganography is mostly craftsmanship, and pretty much by definition cannot be standardised.

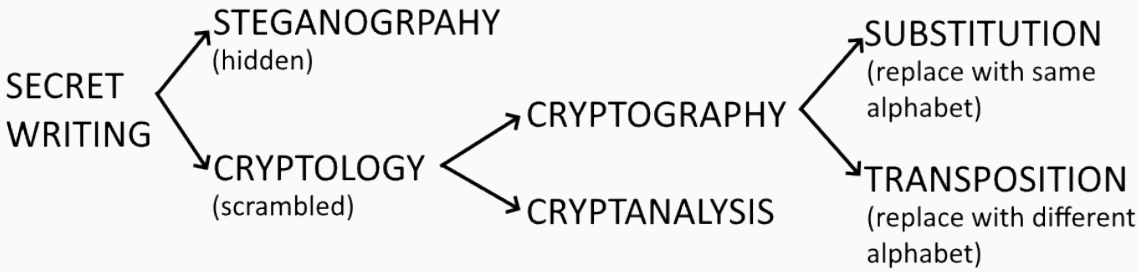
Kerchkhoffs's principle
A good cryptosystem should be secure even though everything, except for the key, is publicly known.

Which is the reason why steganography and cryptography are two different fields.

Cryptography offers much more grounds for theoretical foundations.

Crypto in the Old Days

Monoalphabetic substitution



Monoalphabetic substitution

At least 2000 years old. Used by Greeks, Romans, Indians.

Let $\mathcal{M}$ be the alphabet. Let $\mathcal{K}$ be the set of all permutations of $\mathcal{M}$. For $e \in \mathcal{K}$, we define **monoalphabetic substitution** with the key e by

$$Enc_e(m) = e(m_1)e(m_2)\dots e(m_n) = c$$

Decryption with key e is done similarly:

$$Dec_e(c) = e^{-1}(c_1)e^{-1}(c_2)\dots e^{-1}(c_n) = m$$

Substitution examples

- Caesar: offset by 3 (A→ D, B→ E, ..., W→ Z, X→ A, Y→ B, Z → C)
- ROT13: offset by 13. What is its interesting feature?

How to break it?

How hard is it to break (decrypt) a Caesar cipher?

How to break it?

How hard is it to break (decrypt) a Caesar cipher? 26 possible substitutions. Instant.

In the general case, number of keys for a monoalphabetic substitution:

How to break it?

How hard is it to break (decrypt) a Caesar cipher? 26 possible substitutions. Instant.

In the general case, number of keys for a monoalphabetic substitution: $26! \approx 4.10^{26}$

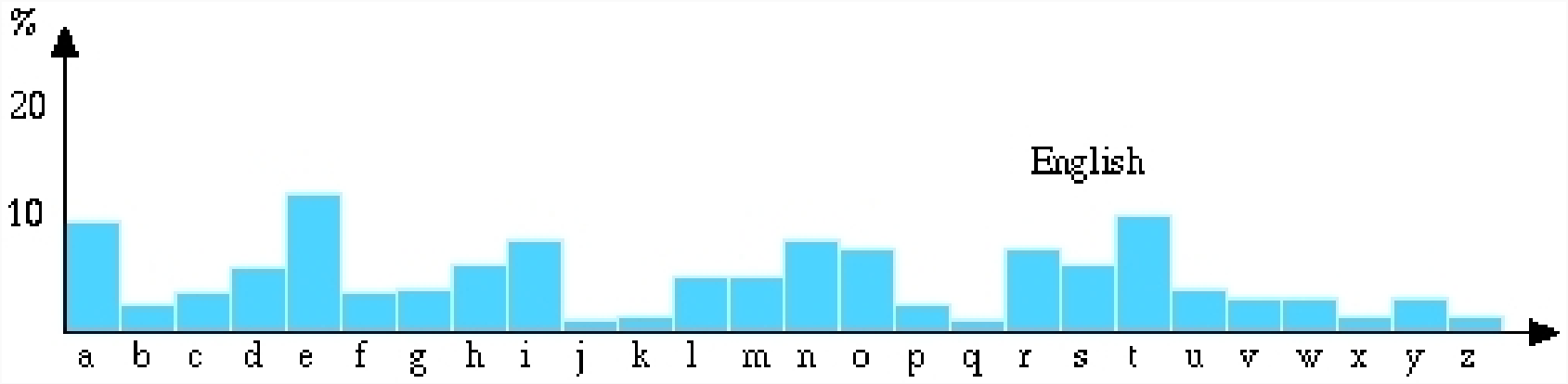
But... ETAOIN SHRDLCU

How to break it?

How hard is it to break (decrypt) a Caesar cipher? 26 possible substitutions. Instant.

In the general case, number of keys for a monoalphabetic substitution: $26! \approx 4.10^{26}$

But... ETAOIN SHRDLCU



How to break a monoalphabetic cipher

- Retrieve the language with index of coincidence
- Do a frequency analysis and retrieve most common letters
- Partially decrypt the ciphertext
- Guess other letters, and iterate.

Poly-alphabetic ciphers

Same.

Poly-alphabetic ciphers

Same. But different.

They try to break the frequency analysis by using different monoalphabetic substitutions, and switching between them.

Most famous is Vigenère cipher. Some variations, such as the one used between the queen Marie-Antoinette and her lover the count Hans Axel von Fersel, also exist.

Vigenère cipher

Let a key $k = k_1 k_2 \dots k_l$. We note by $Enc_e^{mono}(l)$ the monoalphabetic substitution encryption of the letter l with key e.

Vigenère encryption of the message m is
$Enc_e(m) = (Enc_{e_1}^{mono}(m_1))(Enc_{e_2}^{mono}(m_2)) \dots (Enc_{e_{1+(n \mod l)}}^{mono}(m_n))$

Vigenère cipher

Let a key $k = k_1 k_2 \dots k_l$. We note by $Enc_e^{mono}(l)$ the monoalphabetic substitution encryption of the letter l with key e.

Vigenère encryption of the message m is
$Enc_e(m) = (Enc_{e_1}^{mono}(m_1))(Enc_{e_2}^{mono}(m_2)) \dots (Enc_{e_{1+(n \mod l)}}^{mono}(m_n))$

How do we decrypt?

Vigenère cryptanalysis

Simple frequency analysis does not work anymore.

However, repetitions in the ciphertext give an indication of the size of the key.

Then, run a standard monoalphabetic analysis on each class of the quotiented ciphertext.

Beyond Vigenère

The weakness comes from the fact that the key is repeated several times along the message.

What if the key is as long as the message?

What if the key is random?

Modern Crypto

One-time Pad (Vernam cipher)

Let us consider a message $m = m_1 \dots m_n$ written in binary.

Let us consider a random key k, as long as m.

The **One-time pad encryption** of m with key k is $m \oplus k$:

$$Enc_k(m) = (m_1 \oplus k_1) \dots (m_n \oplus k_n)$$

Decryption is simply xoring again: $Dec_k(m) = m \oplus k$.

OTP precisions

- The key must be uniformly randomly selected amongst all possible keys
- A key must not be discarded after one use

Under these conditions, we prove that the one-time pad has **perfect secrecy**: an attacker cannot guess the ciphertext (prove it!).

Hence, we have reached perfect crypto.

Thank you for your attention

Any question?

Kidding.

OTP weaknesses

First, OTP is **malleable**: an attacker can modify a portion of the ciphertext, even without being able to decrypt the ciphertext. (how?)

OTP weaknesses

First, OTP is **malleable**: an attacker can modify a portion of the ciphertext, even without being able to decrypt the ciphertext. (how?)

Also, pads must be unique, and as long as the message. Also truly random, which is not that easy.

Mot importantly, the key must be agreed beforehand by the two parties.

Not practical.

OTP weaknesses

First, OTP is **malleable**: an attacker can modify a portion of the ciphertext, even without being able to decrypt the ciphertext. (how?)

Also, pads must be unique, and as long as the message. Also truly random, which is not that easy.

Mot importantly, the key must be agreed beforehand by the two parties.

Not practical.

However, the OTP was used during WWII, and during the Cold War, securing a line between Moscow and Washington.

Crypto nowadays

Many symmetric systems: AES, Blowfish, Triple DES (deprecated)...

The principle is the same in each case, and can be formalised.

Symmetric cryptosystems

Definition (Symmetric key cryptosystem)
Let $\mathcal{M}$ be the set of all possible messages, $\mathcal{C}$ the set of all possible ciphertexts, and $\mathcal{K}$ the set of all possible keys.

A symmetric cryptosystem is composed of three algorithms:

- KeyGen : $1 \to \mathcal{K}$, the key generator
- $Enc : \mathcal{M} \times \mathcal{K} \to \mathcal{C}$, the encryption function
- $Dec : \mathcal{C} \times \mathcal{K} \to \mathcal{M}$, the decryption function

Furthermore, we must have $Dec(Enc(m, k), k) = m$ for any m and k.

But wait, there's more!

We have described the symmetric scheme: both users share the same key.

Main disadvantages:

- limited functionalities
- $\mathcal{O}(n^2)$ keys for $\mathcal{O}(n)$ participants
- no idea of the actual security of the scheme

So, in the 70s, a new kind of crypto was invented: asymmetric encryption.

Symmetric vs Asymmetric Encryption

Symmetric Encryption (DES, AES)

Asymmetric Encryption (RSA, Elgamal ...)

Brief history of asymmetric crypto

First asymmetric scheme by Merkle in 1974 (published in 1978)

Another scheme by Diffie and Hellman in 1976, still widely in use. Relies on the discrete log problem.

Another star: Rivest, Shamir and Adleman (RSA) in 1978. It relies on the mathematical problem of factorisation.

Secret discovery of RSA by GCHQ (UK secret service) in 1973

Other systems: McEliece (1978, error-correcting codes), Merkle-Hellman (1978, knapsack problem, broken in 1984), ElGamal (1985, discrete log), NTRUEncrypt (1996, lattices), Naccache-Stern (1998, factorisation)...

Asymmetric encryption

Consists of:

- A public key, that you share with everyone else
- A secret key, that you keep for yourself

While symmetric encryption can be thought of as a safe, asymmetric encryption would correspond to a padlock: anyone can close the lock, only one can open it.

Asymmetric encryption

Definition (Asymmetric encryption scheme)
An asymmetric cryptosystem is composed of three algorithms:

- $\text{KeyGen}(1^\lambda) = (\text{sk}, \text{pk})$
- $Enc(m, \text{pk}) = c$
- $Dec(c, \text{sk})$, the decryption function

Furthermore, we must have $Dec(Enc(m, \text{pk}), \text{sk}) = m$ for any m and keypair (pk, sk).

λ is called the security parameter: the higher it is, the more secure the system is.

Asymmetric or symmetric?

Asymmetric crypto is **slow**.

Asymmetric or symmetric?

Asymmetric crypto is **slow**.

However, asymmetric crypto has many advantages, especially when initializing a connection

Hence **hybrid cryptography**: first, contact the target using asymmetric crypto, then agree (under asymmetric crypto) to a secret key for symmetric crypto.

Basic theoretical background

How is it secure?

Behind asymmetric crypto lies the concept of one-way functions.

One-way functions are easy to compute, hard to invert.

How is it secure?

Behind asymmetric crypto lies the concept of one-way functions.

One-way functions are easy to compute, hard to invert.

Fun fact: we do not know whether one-way functions exist (as this would imply $P \neq NP$). But we have good candidates.

Example: factorization

Given two prime numbers, it is easy to compute their product

But given a semi-prime number, it is hard to compute its two prime factors.

Exercise: factor

- 15

Example: factorization

Given two prime numbers, it is easy to compute their product

But given a semi-prime number, it is hard to compute its two prime factors.

Exercise: factor

- 15
- 143

Example: factorization

Given two prime numbers, it is easy to compute their product

But given a semi-prime number, it is hard to compute its two prime factors.

Exercise: factor

- 15
- 143
- a real RSA 2048 key: 25 195 908 475 657 893 494 027 183 240 048 398 571 429 282 126 204 032 027 777 137 836 043 662 020 707 595 556 264 018 525 880 784 406 918 290 641 249 515 082 189 298 559 149 176 184 502 808 489 120 072 844 992 687 392 807 287 776 735 971 418 347 270 261 896 375 014 971 824 691 165 077 613 379 859 095 700 097 330 459 748 808 428 401 797 429 100 642 458 691 817 195 118 746 121 515 172 654 632 282 216 869 987 549 182 422 433 637 259 085 141 865 462 043 576 798 423 387 184 774 447 920 739 934 236 584 823 824 281 198 163 815 010 674 810 451 660 377 306 056 201 619 676 256 133 844 143 603 833 904 414 952 634 432 190 114 657 544 454 178 424 020 924 616 515 723 350 778 707 749 817 125 772 467 962 926 386 356 373 289 912 154 831 438 167 899 885 040 445 364 023 527 381 951 378 636 564 391 212 010 397 122 822 120 720 357 (maybe don't)

OWF, cont'd

One-way functions represent well what we want: easy to encrypt, hard to decrypt

OWF, cont'd

One-way functions represent well what we want: easy to encrypt, hard to decrypt

But it should be easy if we know the secret key!

Such functions are called trapdoors.

The state of the art

The best generic algorithm for factorisation (algebraic sieve) runs heuristically in $\mathcal{O}\left(e^{1.92(\ln n)^{1/3}(\ln \ln n)^{2/3}}\right)$: we call this a sub-exponential complexity.

Better than exponential, but worse than polynomial. Hence, a good enough OWF.

Keysize and security

Remember our λ security parameter?

Keysize and security

Remember our λ security parameter?

$\mathsf{KeyGen}(1^\lambda) = (\mathsf{sk}, \mathsf{pk})$

Keysize and security

Remember our λ security parameter?

$\mathsf{KeyGen}(1^\lambda) = (\mathsf{sk}, \mathsf{pk})$

Estimates for integer factoring Lenstra-Verheul 2000

Modulus (bits)	Operations ($\log_2$)
512	58
1024	80
2048	111
4096	149
8192	156

$\approx 2^{60}$ **years**

According to the level of required security, we will take longer keys.

For factoring-related problems, 2048 bit keys are the official recommendation for now.

Textbook RSA (unsecure)

Select two distinct primes p, q. Compute
$n = pq, \phi(n) = (p-1)(q-1)$.

Find $e < \phi(n)$ coprime with $\phi(n)$, compute $d = e^{-1} \mod \phi(n)$.
Then there is a k such that $ed = k\phi(n) + 1$.

The public key is $\mathsf{pk} = (n, e)$, the secret key is $\mathsf{sk} = d$.

Textbook RSA (unsecure)

Select two distinct primes p, q. Compute
$n = pq, \phi(n) = (p-1)(q-1)$.

Find $e < \phi(n)$ coprime with $\phi(n)$, compute $d = e^{-1} \mod \phi(n)$.
Then there is a k such that $ed = k\phi(n) + 1$.

The public key is $\mathsf{pk} = (n, e)$, the secret key is $\mathsf{sk} = d$.

RSA one-way function: $Enc(m, \mathsf{pk}) = m^e \mod n$

It is assumed that inverting the function (i.e., finding a function $(c = m^e) \mapsto m$) is difficult (**RSA assumption**).

Textbook RSA (unsecure)

Select two distinct primes p, q. Compute $n = pq, \phi(n) = (p-1)(q-1)$.

Find $e < \phi(n)$ coprime with $\phi(n)$, compute $d = e^{-1} \mod \phi(n)$. Then there is a k such that $ed = k\phi(n) + 1$.

The public key is $\mathsf{pk} = (n, e)$, the secret key is $\mathsf{sk} = d$.

RSA one-way function: $Enc(m, \mathsf{pk}) = m^e \mod n$

It is assumed that inverting the function (i.e., finding a function $(c = m^e) \mapsto m$) is difficult (**RSA assumption**).

RSA trapdoor: $Dec(c, \mathsf{sk}) = c^d \mod n$.

This works, because since $c = m^e$, we compute $(m^e)^d = m^{ed} = m^{k\phi(n)+1} = m \mod n$, thanks to Fermat's little theorem.

RSA relies on two assumptions:

- It is hard to factor an integer
- It is hard to compute a modular root

The first one is the one always attacked.

For instance, you can show (do it!) that finding d is equally hard as factoring n

However, it is unclear whereas breaking RSA is equally hard as factoring. It is assumed that not.

Other functionalities in crypto

Encryption is boring

Besides plain encryption, we can do many things:

- Signatures
- Secret sharing
- Zero-knowledge proofs (Goldwasser, Micali, Rackoff, 1985)
- Secure multi-party computation (Yao, 2001)

Encryption is fun again

Interesting crypto techniques have also been discovered:

- Fully Homomorphic encryption
- Searchable encryption
- Format-preserving encryption

Some of them are quite impractical (ciphertexts of several hundreds GB, extremely slow...) and still need heavy research.

Things to remember

- What properties crypto can guarantee
- How Caesar encryption work
- Difference between crypto and stegano
- How Vigenere encryption works
- Everything about one-time pad
- Difference between symmetric and asymmetric encryption
- That crypto is not only about encrypting

Exercises

Decrypt `Jqmv !`, coded with Caesar cipher of offset 8

Decrypt `IVWL FELRIR FMA!`, coded with Vigenere cipher of key `EASY`

Cite three properties that crypto can offer

What is a zero-knowledge proof?

How many keys are there in a public cryptosystem?

Thank you for your attention

Any question?

Next time

- Block ciphers
- ECB/CBC/CTR... modes
- Hashing
- HMAC